Session 131

Scaling Performance Using Grand Central Dispatch and Intel Libraries

Software and Services Group

Richard Hubbard and Pallavi Mehrotra

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Special Thanks

- Phil Kerly, Principal Engineer, Software and Services Group, Intel Corp.

- Justin Landon, Software Engineering Manager, Software and Services Group, Intel Corp.

- Grand Central Dispatch Developers, Apple Inc.

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Our Objectives

- Demonstrate the scalability of Face Tracker application using Apple's new Mac Pro based on Apple's Snow Leopard and Intel® Architecture (code named Nehalem)

- Lessons learned porting Face Tracker application to Apple's Grand Central Dispatch (GCD)

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Agenda

- Revisiting the Intel® Threading Building Blocks (TBB) design
- Grand Central Dispatch design methodology
- Performance results
- Summary

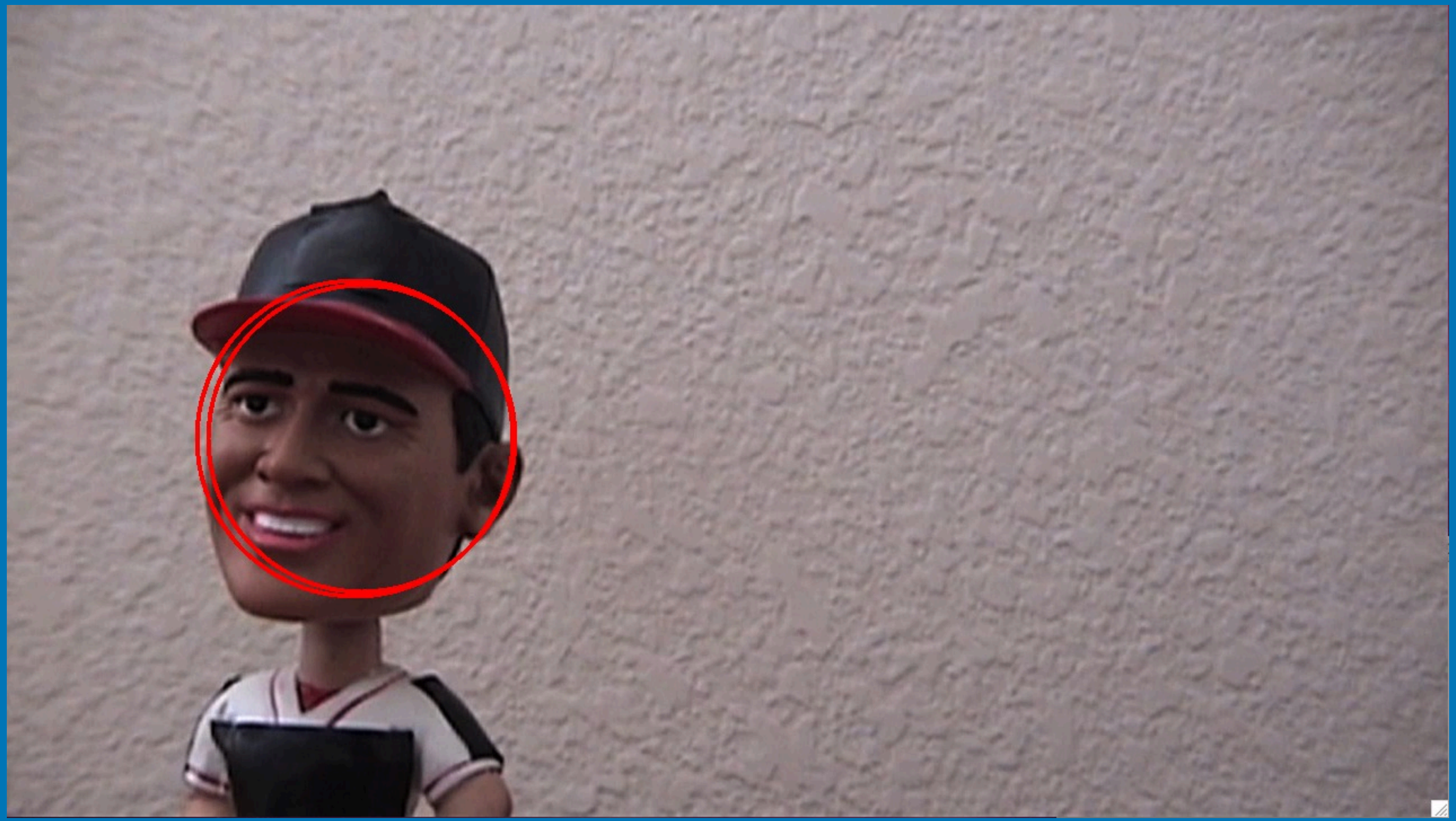
Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Background

- Last year at WWDC 2008
  - We demonstrated a Face Tracker application built by the Intel Compiler which illustrated various threading strategies
    - POSIX* threads
    - OpenMP*
    - Intel® Threading Building Blocks
  - We used the previous generation dual–quad-core Mac Pro
    - 3.2GHz Intel® Xeon® X5482 Processor (Intel Architecture code named “Penryn”)

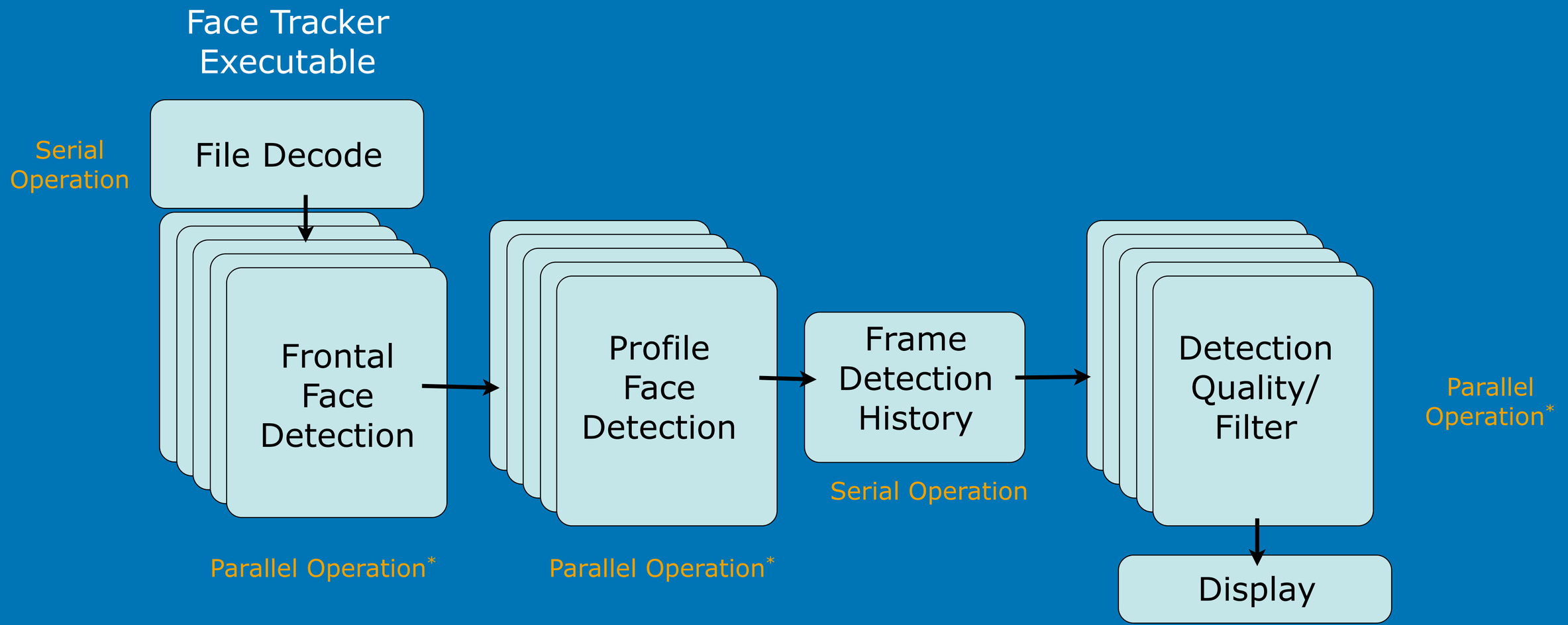
Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Face Tracker Demo



Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

TBB Face Tracker Pipeline



*Separate threads operate on independent frames

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

TBB Pipeline Code—
Face Tracker Class Declaration

```
 // Video Input stage class. Each class is derived from
// “filter” class
class VideoInputFilter: public tbb::filter {
public:
    VideoInputFilter( CvCapture* video_ );
    ~VideoInputFilter();
private:
    /*override*/void* operator()(void*);
};

// Constructor
VideoInputFilter::VideoInputFilter( CvCapture* video_ ) :
// Input filter is a Serial stage, set is_serial as true
filter(/*is_serial=*/true),
video(video_), current_index(0)
{ }
```

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

TBB Pipeline Code—
Face Tracker Work Function

```
// The virtual method filter::operator() inherited from base
// TBB filter class is overridden by the Face Tracker
// class. This method is invoked by the pipeline to
// process an item.
// The parameter item points to the item to be processed.
void* VideoInputFilter::operator()(void*) {

// Process item here....

// The value returned points to the item to be processed
// by the next stage. When no more items need processing,
// first stage of the pipeline should return NULL.
return &frame[current_index];
}
```

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

TBB Pipeline Code—Initializing the Pipeline

```
// Initialize the TBB scheduler. The parameter passed is
// the desired number of threads.
// For optimal results, CORE_NUM is set to the number
// of enabled CPU cores
tbb::task_scheduler_init init(CORE_NUM);

// Create the pipeline
tbb::pipeline pipeline;
```

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

TBB Pipeline Code—
Adding a Stage to the Face Tracker Pipeline

```
// Create and add an input stage to the Face Tracker
// pipeline
VideoInputFilter input_filter( video );
pipeline.add_filter( input_filter );

// Create and add a front face detection stage to the
// Face Tracker pipeline
VideoDetectObjectFilter transform_filter(CASCADE_FRONTAL_NAME,
CV_RGB(0,255,0), true);
pipeline.add_filter( transform_filter );
```

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

TBB Pipeline Code—
Running the Face Tracker Pipeline

```
// Run the TBB Face Tracker pipeline -
// The parameter passed to the run() function controls
// the level of parallelism.
// For optimal results, tbbNumPipeline is set to the
// number of active cpu cores
pipeline.run(tbbNumPipeline);

// Remove filters from pipeline before they are implicitly
// destroyed.
pipeline.clear();
```

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Threading Building Blocks Features

- TBB framework provides high-level abstractions
  - Pipeline
  - Flow-control
  - Reordering between parallel and serial stages
- Portable across platforms—Mac OS X, Windows*, Linux
- Optimized for Intel architecture
- Commercial and open source versions

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Our Key Messages at WWDC 2008…

- Applications need to continually evolve to take advantage of multicore
  - FT demo showed potential for increased video resolutions
  - Added complexity with multiple face tracking algorithms
  - Took advantage of higher level threading optimized modeling libraries like TBB

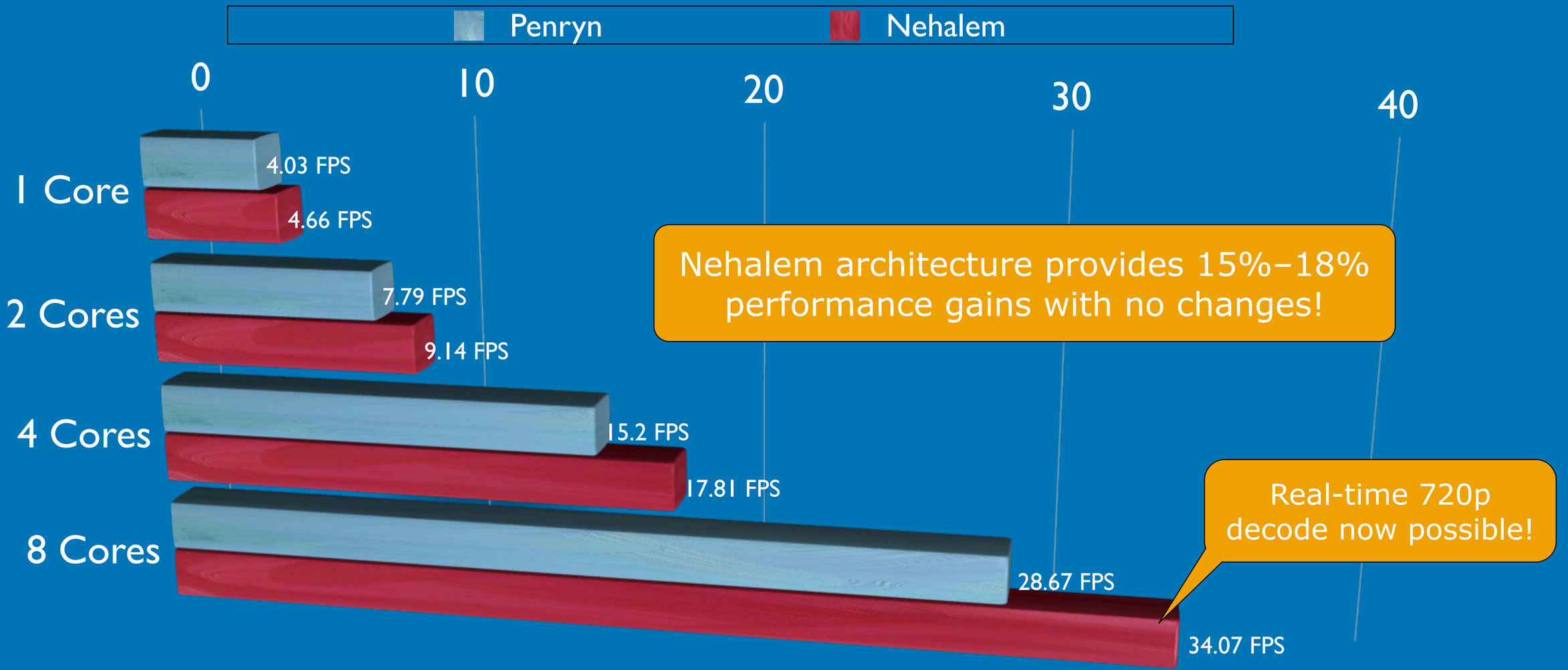
Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Our Key Messages for WWDC 2009

- Illustrate the increased performance the Nehalem architecture brings to software
- Demonstrate Face Tracker on the latest dual–quad-core Mac Pro running Snow Leopard and the Nehalem architecture with Intel® Hyperthreading technology (HT)
- Demonstrate processor scaling with the Face Tracking application ported to Grand Central built with the Intel Compiler

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Boost from Nehalem



720p Face Detection Decode

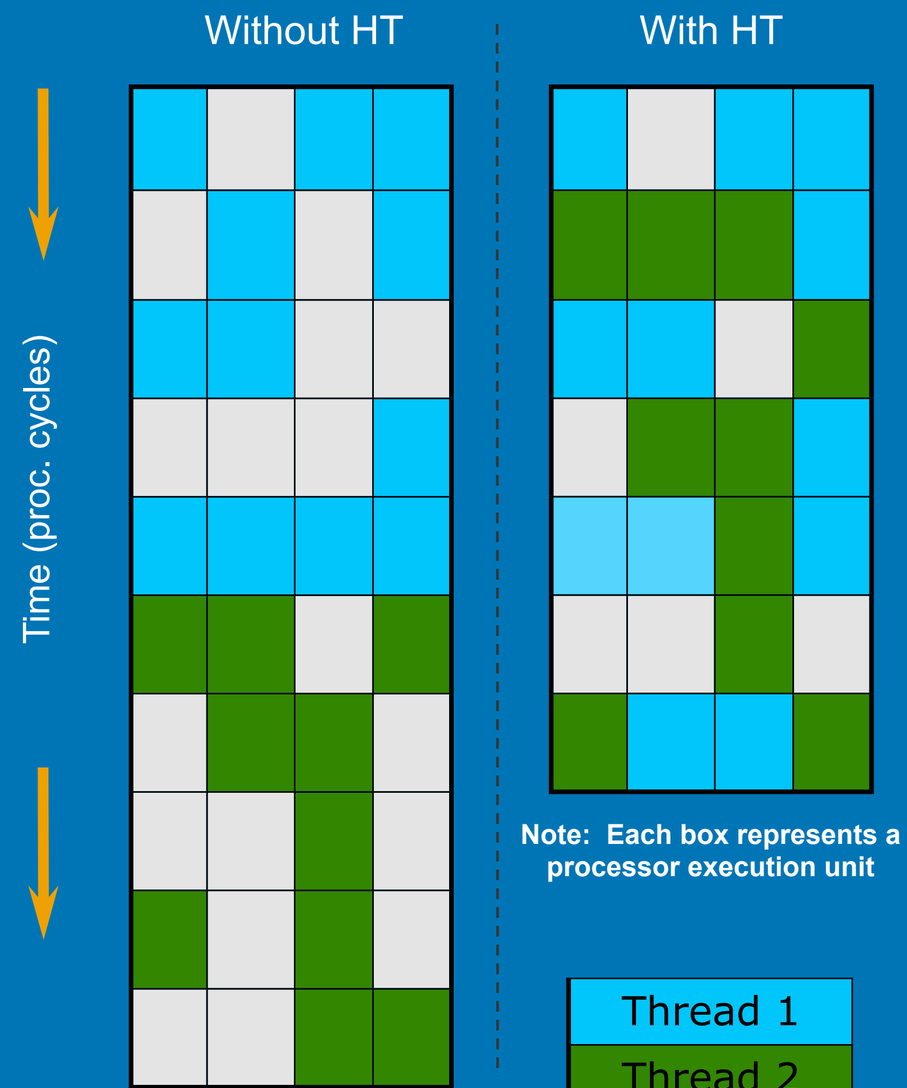
System Configuration
Snow Leopard 10.6 (10A380)
2.93 GHz Intel® Xeon® X5570 Processor v.s 3.2 GHz Intel® Xeon® X5482 Processor
Intel Compiler 11.1.036

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests.Any difference in system hardware or software design or configuration may affect actual performance.Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing.For more information on performance tests and on the performance of Intel products, visit Intel Performance Benchmark Limitations

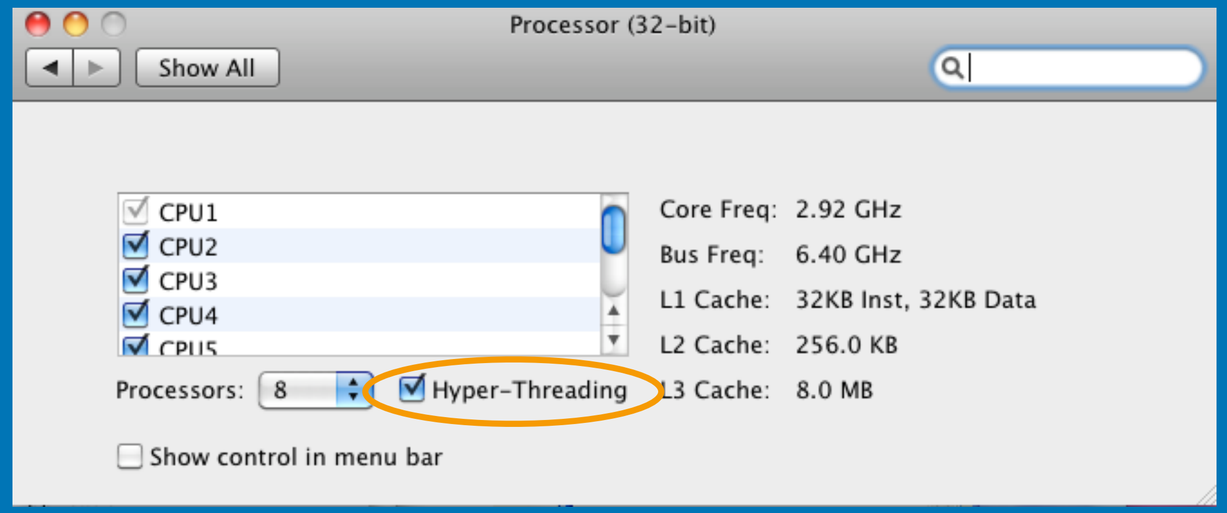
Intel® Hyperthreading Technology

- Hyperthreading (HT)
  - Front-end alternates between two threads per cycle, per core, ensuring fairness
  - Resources are either duplicated, partitioned, or shared
- Take advantage of four-wide execution engine
  - Keep it fed with multiple threads
  - Hide latency of a single thread
- Power efficient
  - Low die area cost, more efficient than adding an entire core
  - Can provide perf speedup depending on app



Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

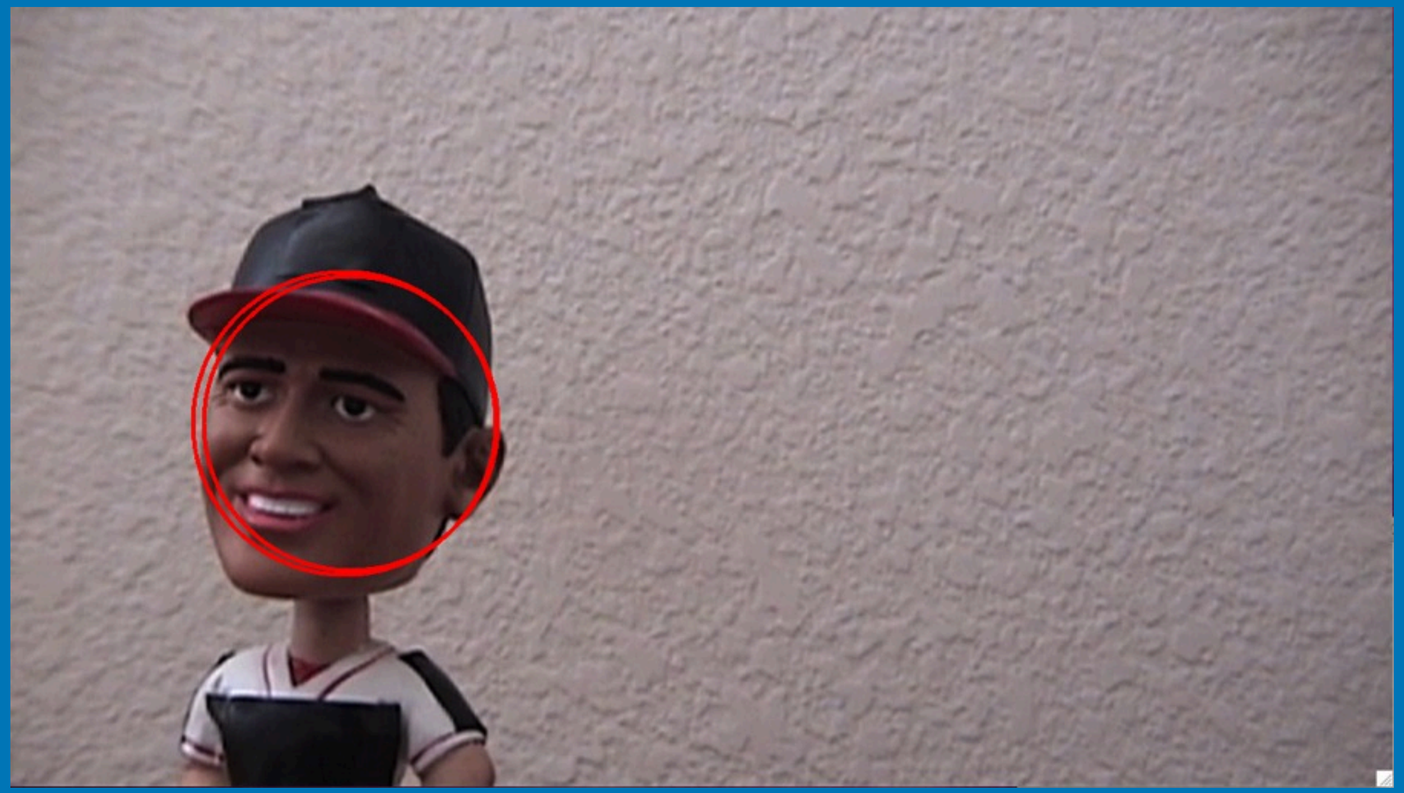
Processor and HT Control via Processor Palette



- Processor pane lets you turn enable/disable HT

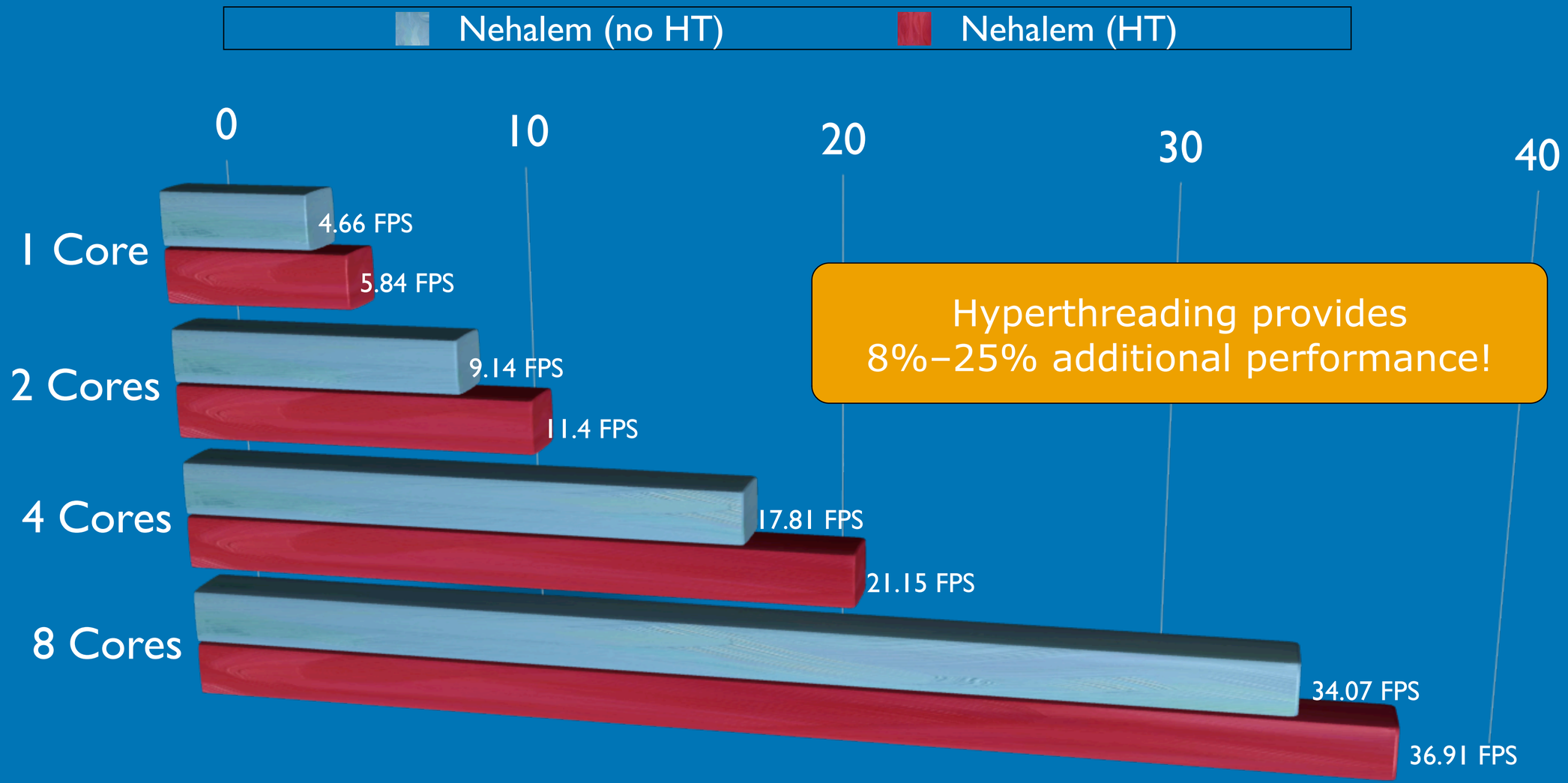
Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

TBB Face Tracker: Eight Cores with HT Demo



Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Additional Performance with HT



720p Face Detection Decode

System Configuration
Snow Leopard 10.6 (10A380)
2.93 GHz Intel® Xeon® X5570 Processor
Intel Compiler 11.1.036

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, visit Intel Performance Benchmark Limitations

TBB Performance

- Summarizing…
  - –Overall performance improved by 29% for 720p
    - Previous generation Mac Pro running Mac OS 10.5 vs. current generation of Mac Pro running Max OS 10.6
    - Support higher resolution clips
      - 720p vs. 420p
  - –Support for increased number of threads
    - Sixteen vs. eight
- Now we'll talk about Grand Central Dispatch

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Porting to Grand Central Dispatch

- What is GCD?
  - Makes Mac OS X multicore aware and optimizes it for allocating tasks across multiple cores and processors
- Goals
  - Learn Apple's latest native threading technology
  - Compare to previous implementations (i.e., TBB)
  - Maintain or improve performance
- Challenges
  - Porting from object-oriented TBB to functional GCD
  - Provide infrastructure
    - Pipelines, flow control, and reordering
- Benefits of Grand Central Dispatch
  - Threads centrally managed by GCD/OS

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Creating the GCD Pipeline

- In TBB…
  - Pipeline is built by the order that objects are added to the pipeline
- In GCD…
  - Face Tracker CStageWorker objects have links to the next stage
  - Links are initialized immediately after object instantiation

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Example: Creating the GCD Pipeline

```
// Build pipeline by instantiating objects and connect them together
// Instantiate stage N first, the last stage, work backwards
VideoOutputStage outputStage(video);

// Instantiate stage N-1
VideoDrawFacesStage
  draw_profile_face_stager_2(CASCADE_PROFILE_NAME, CV_RGB(255,0,0));

// Link stage N-1 to stage N
draw_profile_face_stager_2.nextStage = &outputStage;

// Instantiate stage N-2
VideoDrawFacesStage
  draw_profile_face_stager_1(CASCADE_FRONTAL_NAME, CV_RGB(0,255,0));

// Link stage N-2 to stage N-1
draw_profile_face_stager_1.nextStage = &draw_profile_face_stager_2;
```

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

How Work Traverses the GCD Pipeline

- In TBB…
  - Handled transparently by TBB
- In GCD…
  - CStageWorker objects update context information to next stage
  - Current stage places work on the next stage's queue

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Example: Work Traversing the GCD Pipeline

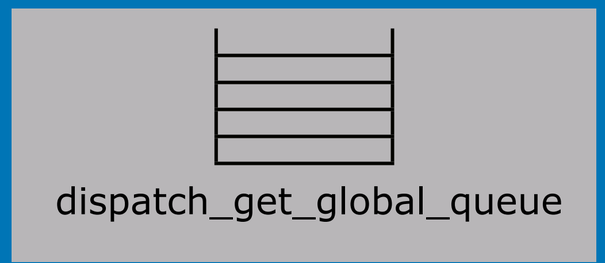
```
// Thin layer for invoking C++ method
void useObjectStage(void * pContext)  {
  ASYNC_CONTEXT * pMyContext = (ASYNC_CONTEXT *) pContext;
  pMyContext->pCStageParent->stageWork(pContext);
}

// Each stage forwards work to the next stage
// Acquire the pointer to the context information
ASYNC_CONTEXT * pMyContext = (ASYNC_CONTEXT *) pContext;

// Acquire the next stage's queue
dispatch_queue_t nextQueue;
nextQueue = **pMyContext->pCStageParent->nextStage->ppQueue;

// Update the stage pointer, next stage
pMyContext->pCStageParent = pMyContext->pCStageParent->nextStage;

// Queue this work
dispatch_async_f(nextQueue, pContext, useObjectStage);
```

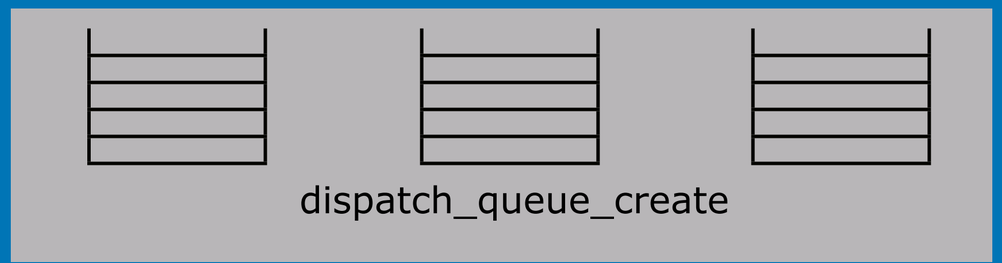
Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Specifying Serial vs. Parallel Stages in GCD

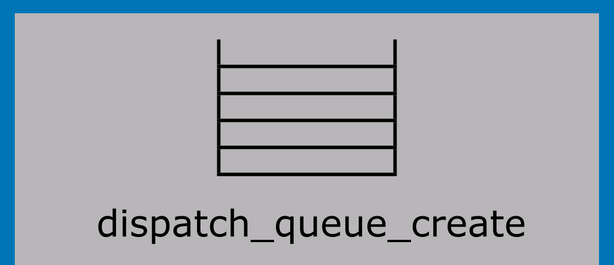
- In TBB…
  - Via parameter to the base class constructor

- In GCD…
  - Parallel: Either global (concurrent) dispatch queue or several FIFO queues





  - Serial: Single FIFO queue



Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Parallel Stages via Multiple GCD Queues

```
// Each stage forwards work to the next stage
// Is the next stage serial or parallel?
dispatch_queue_t nextQueue;
if(pMyContext->pCStageParent->nextStage->queueCount == 1)  {
    nextQueue = **pMyContext->pCStageParent->nextStage->ppQueue;
}
else {

  // Next s
  // Send t
  nextQueue                                                    this-
>queueOffse
  this->que
  this->queueOffset %= pMyContext->pCStageParent->nextStage->queueCount;
}

// Update the stage pointer to the next stage. Queue the work.
pMyContext->pCStageParent = pMyContext->pCStageParent->nextStage;
dispatch_async_f(nextQueue, pContext, useObjectStage);
```

Let GCD Do the Work for You!!

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Detecting Work Completion in GCD

- In TBB…
  - First stage returns NULL when the last frame has been forwarded
- In GCD…
  - First stage checks for last frame
  - Queue a special end-frame directly to output
  - Sets flag to indicate end
  - Wait for a semaphore
    - Work is still flowing through the pipeline

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Example: GCD Work Completion

```
do   {
   // Is this the last frame?
   if (!frame_count){
      bStay = false;
      ASYNC_CONTEXT lastFrameContext;

      lastFrameContext.pCStageParent = &outputStage;
      lastFrameContext.pItem->frameNo = fcount-1;
      lastFrameContext.pItem->lastFrame = FrameInfo::FT_LAST_FRAME;
      dispatch_async_f(**outputStage.ppQueue,  &lastFrameContext,
useObjectStage);
   }
   else  {
      // Standard frame processing
   }
} while (bStay == true);

// Wait for all work to travel through the pipeline
dispatch_semaphore_wait(gcdSemaphore, DISPATCH_TIME_FOREVER);
```

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Objects Place Work on GCD Queues

1. Main() queues face detection work

2. Frontal detect stage queues profile detect work

3. Profile detect stage queues reorder work

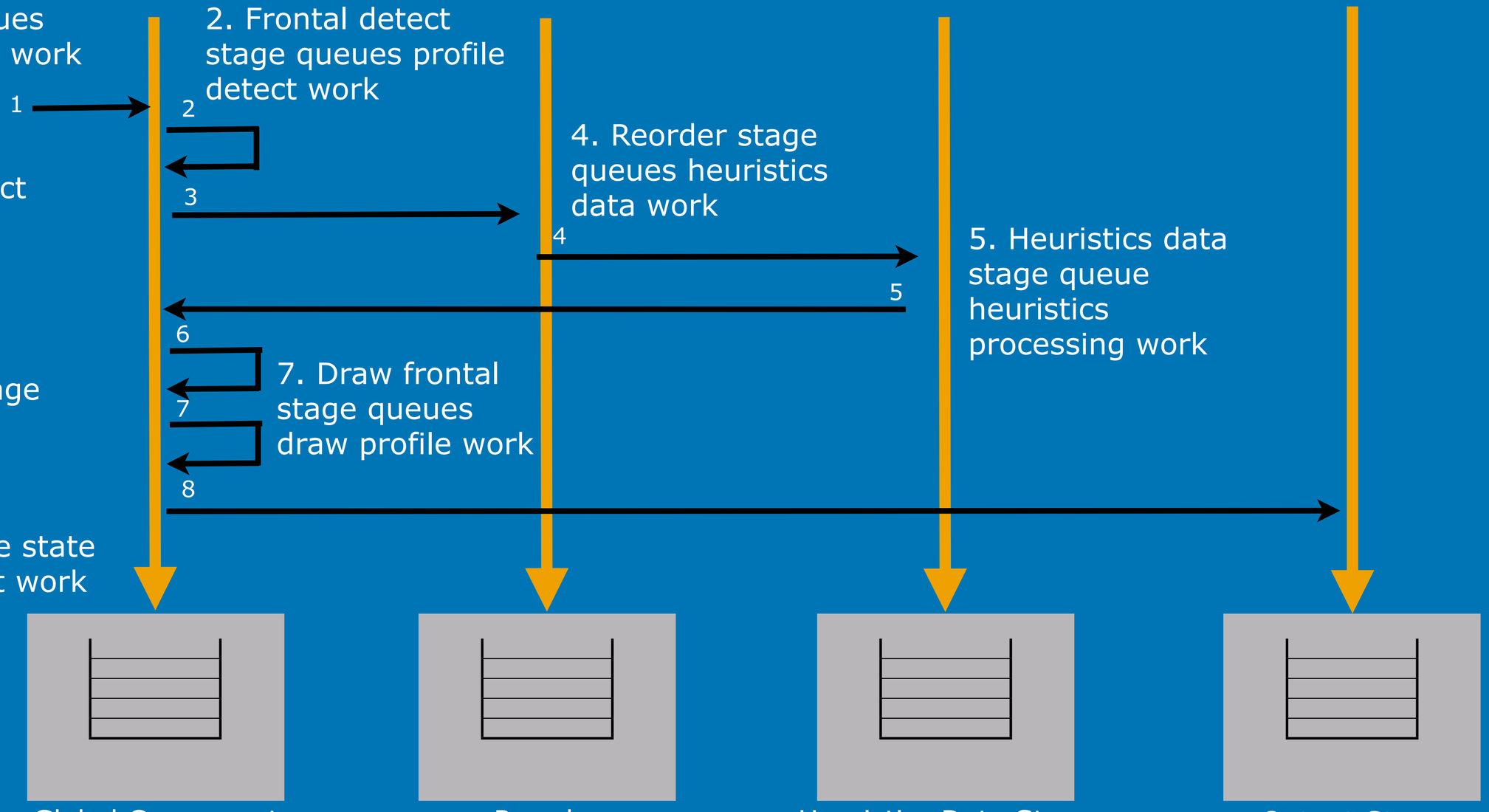
4. Reorder stage queues heuristics data work

5. Heuristics data stage queue heuristics processing work

6. Heuristics processing stage queues draw frontal work

7. Draw frontal stage queues draw profile work

8. Draw profile state queues output work



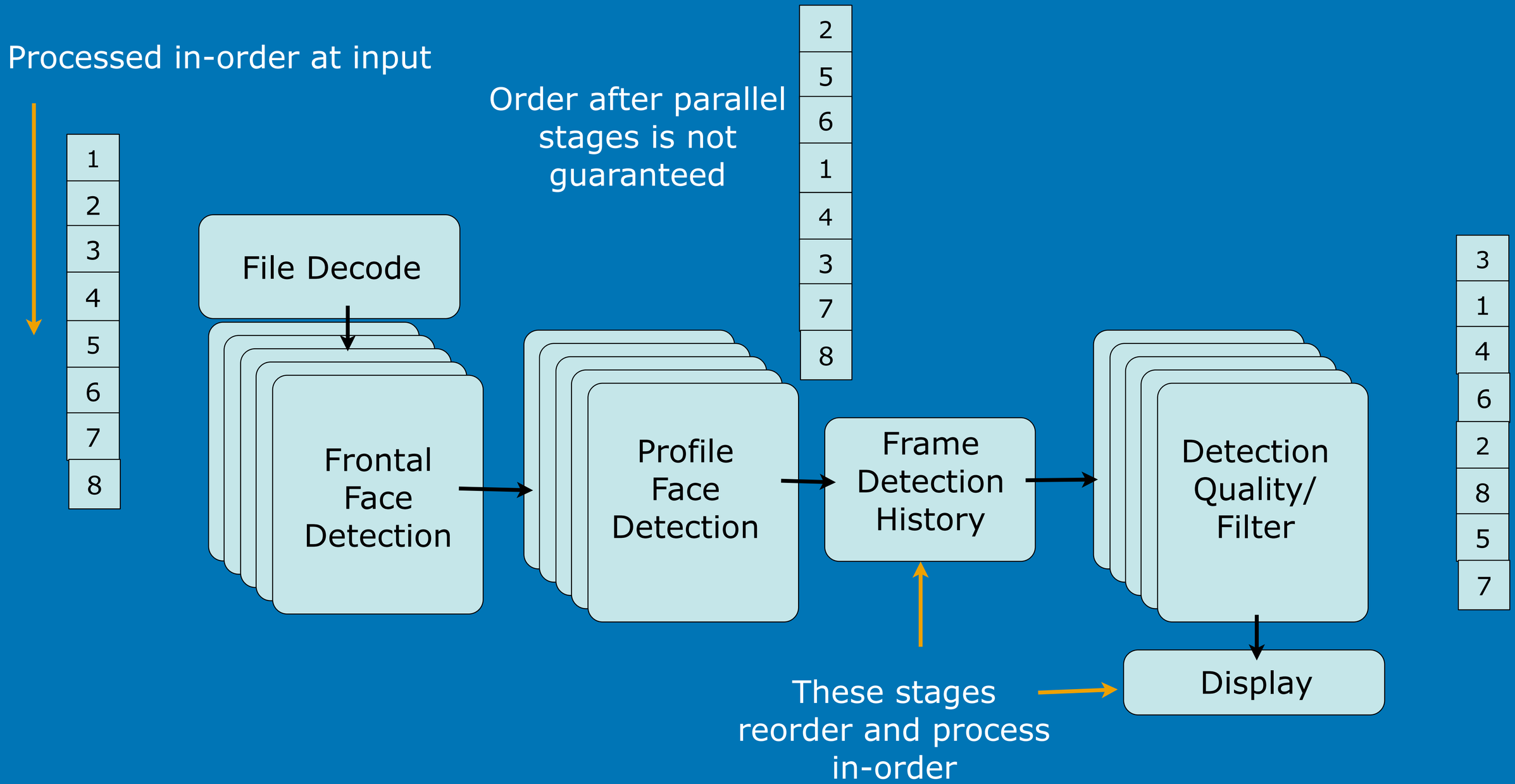
Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Infrastructure for Reordering Frames

- Frames are processed in order at the input stage (main)
- Parallel stages achieve speedup through concurrency
- Output of parallel stages may be out of order
- Simple reorder algorithm
  - Out of order frame contexts are cached
  - Expected frame received
    - All in-order cached frames are processed

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

GCD Reordering

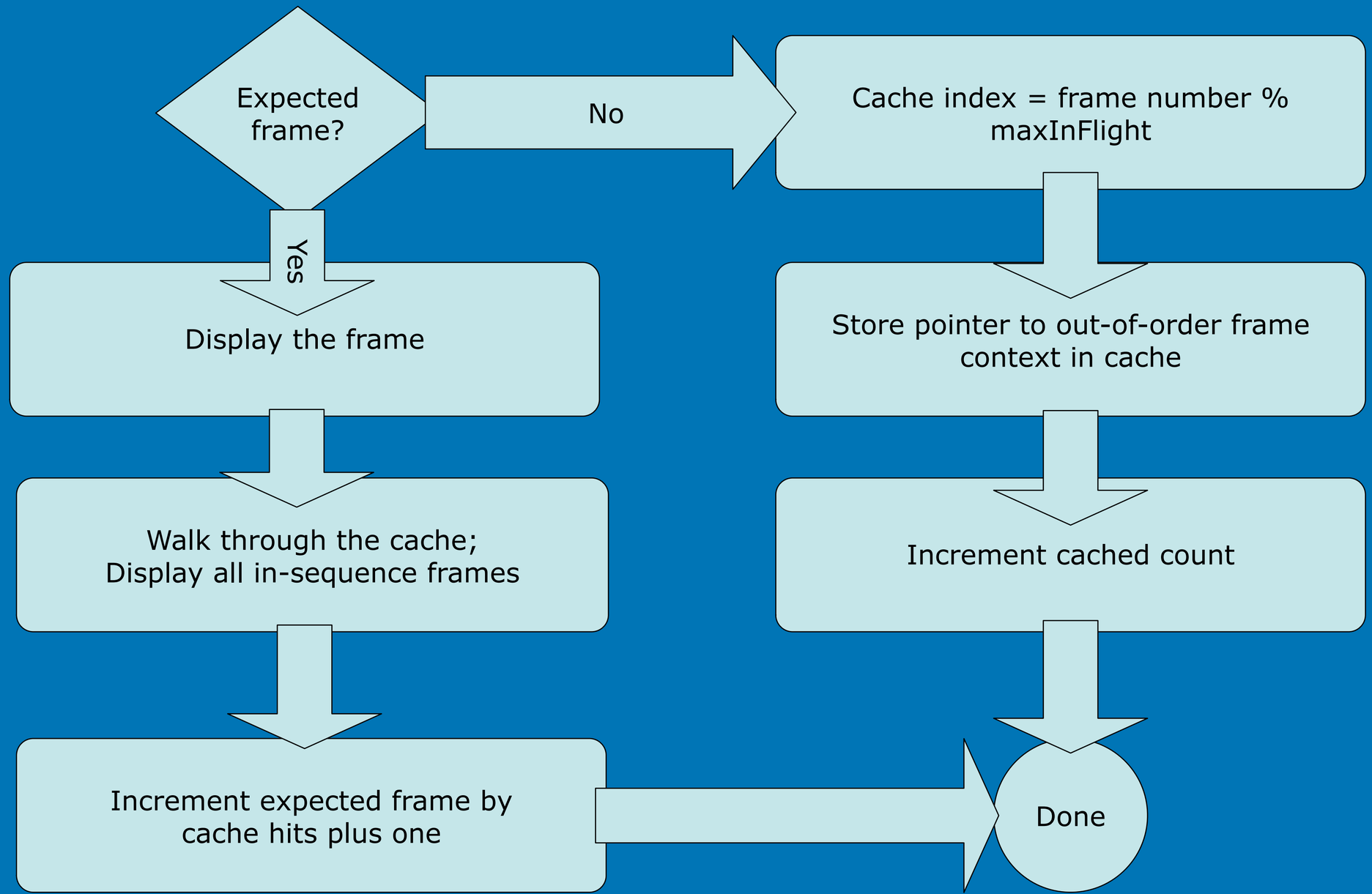


Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

GCD Reordering Frames

In-Order Frame Processing

Out-of-Order Frame Processing



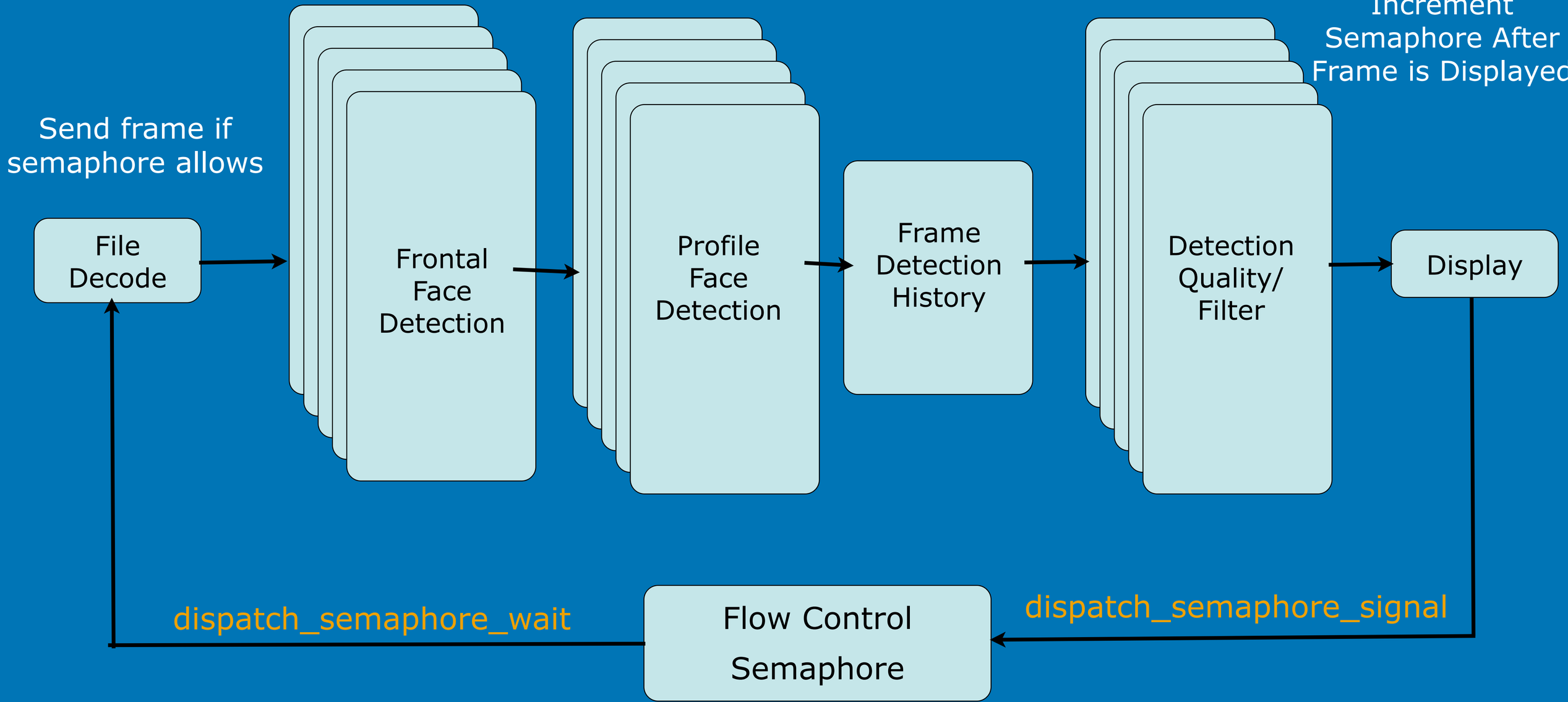
Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

GCD Infrastructure for Flow Control

- For best presentation, limit the number of frames in-flight
  - Flow control between input and output stages
  - Counting semaphore
    - Initialized to maximum number of frames in-flight
    - Input decrements and adds pipeline work
    - Output increments when work is complete
- Signal end of movie via another semaphore

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

GCD Flow Control



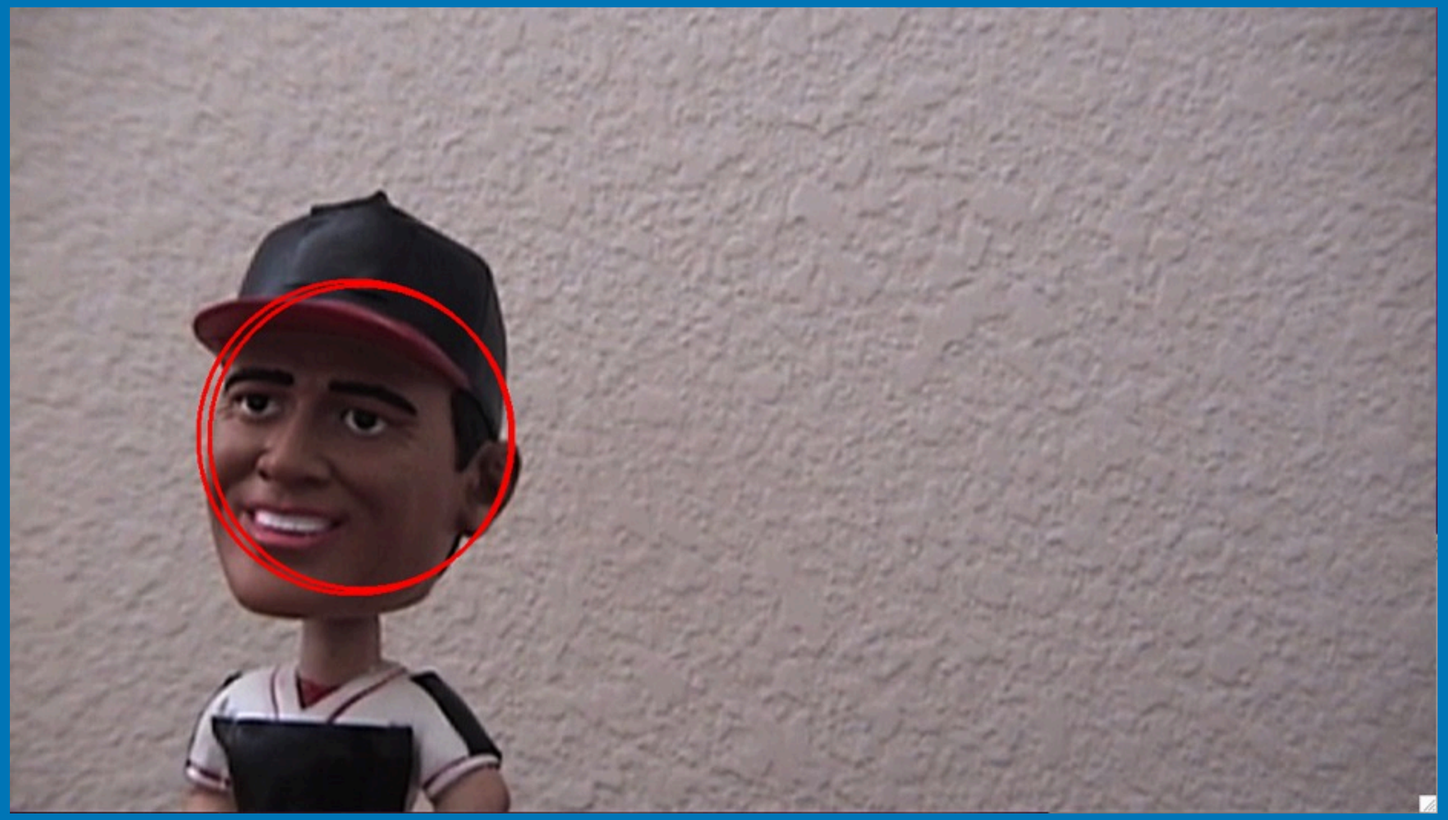
Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

GCD Flow Control

```
// Declare a semaphore for Flow Control
dispatch_semaphore_t g_fcSemaphore;

// Initialize the counting semaphore used for Flow Control
// Maximum in flight at one time equals number of CPUs
g_fcSemaphore = dispatch_semaphore_create(maxInFlight);

// Input Stage Loop
do {
    dispatch_semaphore_wait(g_fcSemaphore, DISPATCH_TIME_FOREVER);

    // Perform input processing on this frame
    ...
} while (bStay == true);

// Output Stage
// After output processing is complete
dispatch_semaphore_signal(g_fcSemaphore)
```

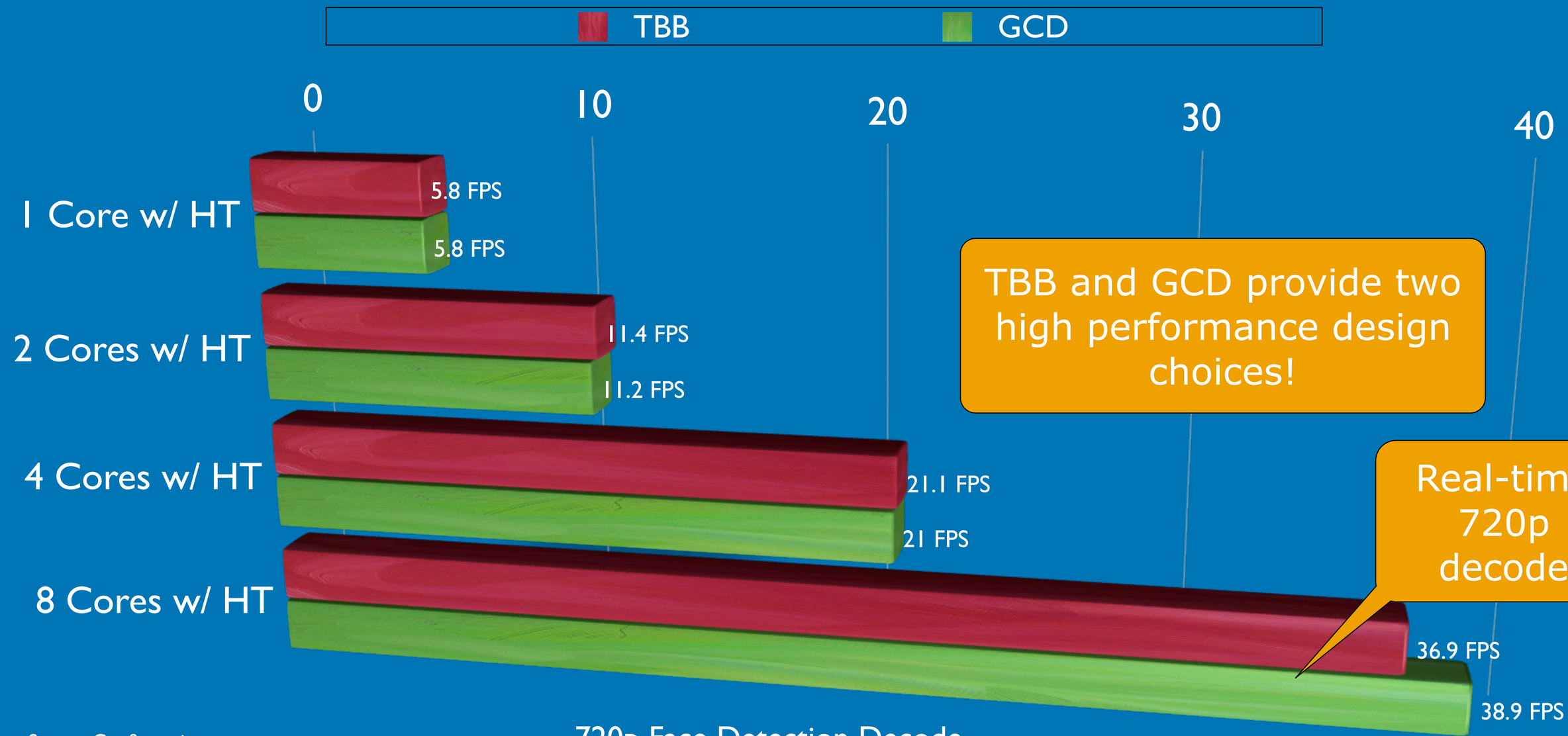
Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

GCD Face Tracker: Eight Cores with HT Demo



Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Comparing GCD and TBB Performance



TBB and GCD provide two high performance design choices!

Real-time 720p decode

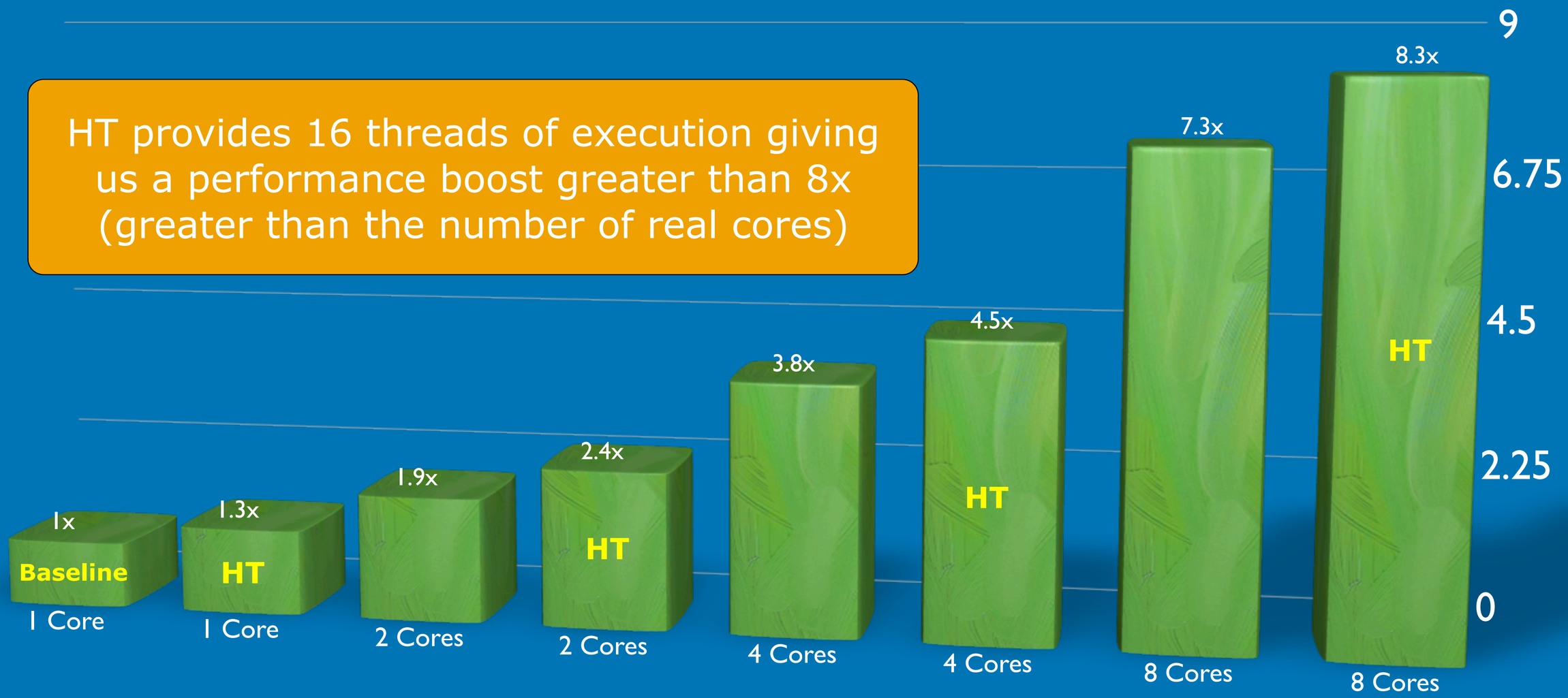
System Configuration
Snow Leopard 10.6 (10A380)
2.93 GHz Intel® Xeon® X5570 Processor
Intel Compiler 11.1.036

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, visit Intel Performance Benchmark Limitations

GCD Processor Scaling

HT provides 16 threads of execution giving us a performance boost greater than 8x (greater than the number of real cores)



System Configuration
Snow Leopard 10.6 (10A380)
2.93 GHz Intel® Xeon® X5570 Processor
Intel Compiler 11.1.036

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, visit Intel Performance Benchmark Limitations

Key GCD Developer Takeaways

- Centralized thread management
- Construct pipelines to fit your needs
- Let GCD do the work for you
  - Arrays of queues requires management
  - Global queues removes this burden
- Intel Compiler does not currently support Block Objects
  - Use function pointer versions of GCD APIs with no performance penalty
- Standard debugging techniques still apply

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Performance Summary

- Snow Leopard and Nehalem Architecture provide a winning combination
  - Nehalem provided 15–18% performance gains
  - Hyperthreading provided 8–25% performance gains
- TBB and GCD provide two high performance design choices
- Original key messages still true today as they were a year ago
  - More features/complexity
  - Use technologies like TBB and GCD to improve scalability

It’s been a great year for Mac Pro performance!

Copyright © 2009, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners
Intel, the Intel logo, Itanium, Pentium and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

intel®
Leap ahead™